d77 User Manual

Shizu Katsuyuki (志津　功將)

2024

1 License

d77 is free software and can be redistributed and/or modified under the terms of the GNU General Public License v3.0 as published by the Free Software Foundation. For details, see https://www.gnu.org/licenses/gpl-3.0.html.

2 Citation

You can cite d77 as

d77: visual understanding of molecular properties. Version 20240427. https://github.com/KatsuyukiShizu/d77

BibTex description:

```
@misc{d77,
title = {
d77: visual understanding of molecular properties.
Version 20240427. https://github.com/KatsuyukiShizu/d77
}
}
```

3 Notations

PGF	primitive Gaussian function
CGF	contracted Gaussian function
VCC	vibronic coupling constant
SOC	spin-orbit coupling
fchk file	formatted checkpoint file
TD-DFT	time-dependent density functional theory
CI	configuration interaction
CIS	configuration interaction singles
ρ_n	electron density of electronic state $\lvert n\rangle$
ρ_{mn}	overlap (transition) density between electronic states $\lvert m\rangle$ and $\lvert n\rangle$
$\Delta\rho_{mn}$	electron-density difference: $\rho_m - \rho_n$
$\boldsymbol{\mu}$	permanent/transition dipole moment
τ_x	permanent/transition dipole moment density in the x direction
τ_y	permanent/transition dipole moment density in the y direction
τ_z	permanent/transition dipole moment density in the z direction
$\tau_{\boldsymbol{\mu}}$	permanent/transition dipole moment density in the direction of $\boldsymbol{\mu}$
ν_k	derivative of the nuclear electronic potential for the kth vibrational mode
η_k	vibronic coupling density for the kth vibrational mode

4 Compiling and running d77

You can download Fortran90 source files, documents, and examples from GitHub:[1]
https://github.com/KatsuyukiShizu/d77
Shell scripts (d77.sh and run_d77.sh), Fortran90 source files, and Makefile are in the src directory. Go to your src directory and type:

```
make
```

The executable file (d77.exe) is created in your src directory.

The default settings for the maximum number of electrons (Max_ne), the maximum number of atoms (Max_n_atm), the maximum number of vibrational modes (Max_n_mode), the maximum number of CGFs (Max_n_cgf), the maximum number of PGFs for each CGF (Max_n_pgf), and the maximum number of electronic configurations (Max_n_elec_config) are defined in a module, global_constants.f90:

```
Max_ne = 100
Max_n_atm = 30
Max_n_mode = 100
Max_n_cgf = 1000
Max_n_pgf = 30
Max_n_state = 20
Max_n_elec_config = 1000
```

Change the default settings depending on your computer's system environment.

Utility programs for d77 are in the src/util directory. A utility program, dpp, performs data preprocessing for d77. dpp extracts data from Gaussian 16 output files (fchk and log files) and saves them to INP_ELEC and/or INP_VIB directory. Shell scripts (dpp.sh and others) are in the src/util/dpp directory. Fortran90 source files and Makefile are in the src/util/dpp/f90 directory. Go to your src/util/dpp/f90 directory and type:

```
make
```

The executable file (dpp.exe) is created in your src/util/dpp/f90 directory.

Edit your configuration file, for example, .bashrc:

```
alias d77='sh ABCD/d77/src/d77.sh'
export DIR_d77_EXE=ABCD/d77/src
alias dpp='sh ABCD/d77/src/util/dpp/dpp.sh'
export DIR_d77_dpp_f90=ABCD/d77/src/util/dpp/f90
```

Here, ABCD is a directory where you created d77 directory. To run d77, simply type:

```
d77 'input file' 'results directory'
```

If you have an input file, *test.inp*, and want to save the log file (*test.log*) and calculated data to *res_test* directory, type:

```
d77 test.inp res_test
```

or

```
d77 test res_test
```

You can run the job without specifying a results directory:

```
d77 test
```

In this case, the log file and calculated data are saved to *test* directory.

5 Input file description

5.1 Comment lines

Add comments between $comment and $end_comment lines in your input file. Comment lines have no influence on d77 execution.

5.2 Control options

Add control options between $control and $end_control lines.

`property`

dipole	calculates permanent/transition dipole moment (default).
vc	calculates diagonal/off-diagonal VCC.
soc	calculates SOC between singlet and triplet states.
rho	calculates electron density or overlap (transition) density if runtyp = density.

`runtyp`

calc_int_cgf	calculates one-electron integrals between CGFs with the McMurchie and Davidson formulation.[2]
int_pgf	calculates molecular properties from one-electron integrals between PGFs with the McMurchie and Davidson formulation[2] (default).
density	calculates molecular property densities and writes them to cube files.
cube	manipulates existing cube files and generates a new cube file.

`method`

td	reads **X** and **Y** calculated with TD-DFT (default).
cis	reads CI coefficients calculated with CIS.

`effcharg_soc`

read	reads effective nuclear charges from ZEFF_SOC file (default).
koseki	uses Koseki's effective nuclear charges.[3–5]
nuccharg	uses nuclear charges as effective nuclear charges for SOC calculation.

`cube_op`

add	adds two cube files and saves the calculated date to a new cube file.
sub	subtracts two cube files and saves the calculated date to a new cube file.
mul	multiplies two cube files and saves the calculated date to a new cube file.

5.3 Spin-orbit coupling options

Add spin-orbit coupling options between $spin_orbit and $end_spin_orbit lines.

`ms`

0	calculates SOC between singlet and triplet states with $M_\mathrm{s} = 0$ (default).
1	calculates SOC between singlet and triplet states with $M_\mathrm{s} = 1$.
-1	calculates SOC between singlet and triplet states with $M_\mathrm{s} = -1$.

5.4 Electronic state options

Add electronic state options between $elec_state and $end_elec_state lines.

`bra`

$\langle 0 \vert$	ground state (default)
$\langle n \vert$	nth excited state ($n \geq 1$)

`ket`

$\vert 0 \rangle$	ground state (default)
$\vert n \rangle$	nth excited state ($n \geq 1$)

The following description calculates molecular properties between the first and third excited states:

```
$elec_state
bra = < 1 |
ket = | 3 >
$end_elec_state
```

5.5 Grid options

Grid options are valid if runtyp = density. Add the grid options between $grid and $end_grid lines.

xmin	minimum x coordinate in bohr
xmax	maximum x coordinate in bohr
ymin	minimum y coordinate in bohr
ymax	maximum y coordinate in bohr
zmin	minimum z coordinate in bohr
zmax	maximum z coordinate in bohr
dx	grid spacing in x direction (the default value is 0.25 bohr)
dy	grid spacing in y direction (the default value is 0.25 bohr)
dz	grid spacing in z direction (the default value is 0.25 bohr)

5.6 Input data directories

Add input data directories in your input file. If your INP_ELEC, INP_VIB, ELFLD, and SOC_CGF directories are in the same directory where your input file is, add the following descriptions:

```
INPDIR_ELEC = INP_ELEC
INPDIR_VIB = INP_VIB
INPDIR_ELFLD = ELFLD
INPDIR_SOC_CGF = SOC_CGF
```

6 Examples

6.1 VCCs between singlet/triplet states at optimized S_0 geometry

This example shows how VCCs are calculated with d77. Sample d77 input and output files are in the examples/C2H4 and examples/C2H4/VC directories.

1. Run geometry optimization and frequency analysis with Gaussian 16.[6] 6D and 10F options are required to run d77.

   (a) Input description for geometry optimization

   ```
   #P B3LYP/6-31G(d)
   6D 10F
   opt
   ```

   (b) Input description for frequency analysis

   ```
   %chk=freq.chk
   #P B3LYP/6-31G(d)
   6D 10F
   Freq=HPModes
   ```

   (c) Input description for excited-state calculation with TD-DFT

   ```
   %chk=TD.chk
   #P B3LYP/6-31G(d)
   6D 10F
   TD(50-50, Nstates=5)
   IOP(9/40=4)
   ```

2. Generate fchk files from the checkpoint files. Use the formchk utility of the Gaussian 16 program:

   ```
   formchk freq.chk
   formchk TD.chk
   ```

   Confirm that freq.fchk and TD.fchk files are generated. Sample freq.fchk, TD.fchk, and TD.log files are in the examples/C2H4 directory.

3. Create an input file (dpp.inp) for data preprocessing and run dpp. dpp.inp is formatted as follows:

   ```
   QC_PROGRAM_ELEC = g16
   QC_PROGRAM_VIB  = g16
   QC_METHOD = td
   PROPERTY = vc

   THRESHOLD_CICOEF = 1.0D-4

   DATA_ELEC_MODELSYS = TD.fchk
   DATA_CICOEF_MODELSYS = TD.log
   DATA_VIB_REALSYS = freq.fchk
   ```

   To run dpp, type:

```
dpp
```

INP_ELEC and INP_VIB directories are created. Sample INP_ELEC and INP_VIB directories are in the examples/C2H4 directory.

4. Calculate electric field integrals between CGFs with d77.
   A sample input file (ELFLD.inp) is as follows:

   ```
   $comment
   Input file for calculating electric field integrals between CGFs
   $end_comment

   $control
   property = vc
   runtyp   = calc_int_cgf
   $end_control

   DIR_INP_ELEC = INP_ELEC
   ```

   To run d77, type:

   ```
   d77 ELFLD
   ```

   d77 reads information on the nuclear coordinates and atomic orbitals from INP_ELEC directory, calculates the electric field integrals, and writes them to a file (ELFLD_CGF_ATM) in the ELFLD directory. Note that ELFLD_CGF_ATM takes up large amounts of disk space (sometimes larger than 10 GB) for large molecules. A sample ELFLD_CGF_ATM file is in the examples/C2H4/ELFLD directory.

5. Calculate VCCs between S_0 and S_1 from the electric field integrals.
   Create VC directory and move into it:

   ```
   mkdir VC
   cd VC
   ```

   A sample input file (VC_0_3.inp) is as follows:

   ```
   $comment
   Input file for calculating vibronic coupling constants
   between S0 and S1
   | 0 > : ground state (S0)
   | 3 > : lowest excited singlet state (S1)
   $end_comment

   $control
   property = vc
   runtyp   = int_pgf
   method   = td
   $end_control

   $elec_state
   bra = < 0 |
   ket = | 3 >
   ```

```
$end_elec_state

DIR_INP_ELEC  = ../INP_ELEC
DIR_INP_VIB   = ../INP_VIB
DIR_INP_ELFLD = ../ELFLD
```

To run d77, type:

```
d77 VC_0_3
```

The frequencies (in cm^{-1}) and calculated S_0-S_1 VCCs (in atomic units) are written in VC_0_3.log and VCC files in VC_0_3 directory. The transition dipole moments between S_0 and S_1 are also written in VC_0_3.log (in atomic units and debye) and DM files (in atomic units) in the VC_0_3 directory.

6. Calculate VCCs between T_1 and T_2 from the electric field integrals.
   A sample input file (VC_1_2.inp) is as follows:

```
$comment
Input file for calculating vibronic coupling constants
between T1 and T2
| 1 > : lowest triplet state (T1)
| 2 > : second lowest triplet state (T2)
$comment

$control
property = vc
runtyp   = int_pgf
method   = td
$end_control

$elec_state
bra = < 1 |
ket = | 2 >
$end_elec_state

DIR_INP_ELEC  = ../INP_ELEC
DIR_INP_VIB   = ../INP_VIB
DIR_INP_ELFLD = ../ELFLD
```

To run d77, type:

```
d77 VC_1_2
```

6.2 SOCs between singlet and triplet states at optimized S_0 geometry

This example shows how SOCs are calculated with d77. Sample d77 input and output files are in the examples/C2H4/SOC directories. The first three steps are the same as in **6.1**.

1. Run geometry optimization and frequency analysis with Gaussian 16. 6D and 10F options are required to run d77.

2. Generate freq.fchk and TD.fchk files.

3. Run dpp.

4. Calculate SOC integrals between CGFs with d77.
   A sample input file (SOC_CGF.inp) is as follows:

   ```
   $comment
   Input file for calculating spin-orbit integrals between CGFs
   $end_comment

   $control
   property = soc
   runtyp   = calc_int_cgf
   $end_control

   DIR_INP_ELEC = INP_ELEC
   ```

   To run d77, type:

   ```
   d77 SOC_CGF
   ```

   d77 reads information on the nuclear coordinates and atomic orbitals from the INP_ELEC directroy, calculates the spin-orbit integrals, and writes them to files (SOC_CGF_X, SOC_CGF_Y, and SOC_CGF_Z) in the SOC_CGF directory.

5. Calculate x, y, and z components of SOC between S_0 and T_1 with $M_\mathrm{s} = 0$ from the SOC integrals between CGFs.
   Create SOC directory and move into it:

   ```
   mkdir SOC
   cd SOC
   ```

   A sample input file (SOC_0_1_Ms_0.inp) is as follows:

   ```
   $comment
   Input file for calculating spin-orbit coupling
   between S0 and T1 (Ms = 0)
   | 0 > : ground state (S0)
   | 1 > : lowest triplet state (T1)
   $end_comment

   $control
   property     = soc
   runtyp       = int_pgf
   method       = td
   effcharg_soc = read
   $end_control

   $spin_orbit
   ms = 0
   $end_spin_orbit

   $elec_state
   bra = < 0 |
   ```

```
ket = | 1 >
$end_elec_state

DIR_INP_ELEC    = ../INP_ELEC
DIR_INP_SOC_CGF = ../SOC_CGF
```

To run d77, type:

```
d77 SOC_0_1_Ms_0
```

The calculated x, y, and z components of S_0-T_1 SOC are written in SOC_0_1_Ms_0.log (atomic units and cm^{-1}) and SOC (atomic units) files in the SOC_0_1_Ms_0 directory.

6. Calculate x, y, and z components of SOC between S_0 and T_1 with $M_s = 1$ ($M_s = -1$). Replacing ms = 0 with ms = 1 (ms = −1) gives an input file for calculating the x, y, and z components of the SOC between S_0 and T_1 with $M_s = 1$ ($M_s = -1$).

7. Calculate x, y, and z components of S_0-T_2 SOC with $M_s = 0$. A sample input file (SOC_0_2_Ms_0.inp) is as follows:

```
$comment
Input file for calculating spin-orbit coupling
between S0 and T2 (Ms = 0)
| 0 > : ground state (S0)
| 2 > : second lowest triplet state (T2)
$end_comment

$control
property     = soc
runtyp       = int_pgf
method       = td
effcharg_soc = read
$end_control

$spin_orbit
ms = 0
$end_spin_orbit

$elec_state
bra = < 0 |
ket = | 2 >
$end_elec_state

DIR_INP_ELEC    = ../INP_ELEC
DIR_INP_SOC_CGF = ../SOC_CGF
```

To run d77, type:

```
d77 SOC_0_2_Ms_0
```

6.3 Electron density, electron-density difference (difference electron-density; electron-difference density), and overlap (transition) density at optimized S_0 geometry

This example shows how electron density and overlap (transition) density are calculated with d77. Sample d77 input and output files are in the examples/C2H4/RHO directories. The first three steps are the same as in **6.1**.

1. Run geometry optimization and frequency analysis with Gaussian 16. 6D and 10F options are required to run d77.
2. Generate freq.fchk and TD.fchk files.
3. Run dpp.
4. Calculate electron density of S_0 (ρ_{S_0}) with d77.
   Create RHO directory and move into it:

   ```
   mkdir RHO
   cd RHO
   ```

   A sample input file (RHO_0_0.inp) is as follows:

   ```
   $comment
   Input file for calculating electron density of S0
   | 0 > : ground state (S0)
   $end_comment

   $control
   property = rho
   runtyp   = density
   method   = td
   $end_control

   $elec_state
   bra = < 0 |
   ket = | 0 >
   $end_elec_state

   $grid
   xmin = -5.0
   xmax =  5.0
   ymin = -5.0
   ymax =  5.0
   zmin = -5.0
   zmax =  5.0
   dx   = 0.20
   dy   = 0.20
   dz   = 0.20
   $end_grid

   DIR_INP_ELEC = ../INP_ELEC
   ```

   To run d77, type:

```
d77 RHO_0_0
```

d77 calculates the electron density of S_0 and writes it to a cube file (RHO.cube) in the RHO_0_0 directory. You can visualize RHO.cube using Chemcraft[7] or GaussView 6,[8] for example.

5. Calculate electron density of S_1 (ρ_{S_1}).
   A sample input file (RHO_3_3.inp) is as follows:

```
$comment
Input file for calculating electron density of S1
| 3 > : lowest excited singlet state (S1)
$end_comment

$control
property = rho
runtyp   = density
method   = td
$end_control

$elec_state
bra = < 3 |
ket = | 3 >
$end_elec_state

$grid
xmin = -5.0
xmax =  5.0
ymin = -5.0
ymax =  5.0
zmin = -5.0
zmax =  5.0
dx   = 0.20
dy   = 0.20
dz   = 0.20
$end_grid

DIR_INP_ELEC = ../INP_ELEC
```

   To run d77, type:

```
d77 RHO_3_3
```

6. Calculate electron-density difference $\Delta\rho_{\mathrm{S}_1\mathrm{S}_0}$ ($= \rho_{\mathrm{S}_1} - \rho_{\mathrm{S}_0}$).
   A sample input file (DIFF_DENSITY.inp) is as follows:

```
$comment
Input file for calculating electron density difference
between S0 and S1: (S1 electron density) - (S0 electron density)
| 0 > : ground state (S0)
| 3 > : lowest excited singlet state (S1)
$end_comment
```

```
$control
runtyp  = cube
cube_op = sub
$end_control

CUBE_RES = DIFF_DENSITY_S1_S0.cube
CUBE_1   = RHO_3_3/RHO.cube
CUBE_2   = RHO_0_0/RHO.cube
```

To run d77, type:

```
d77 DIFF_DENSITY
```

The calculated $\Delta\rho_{\mathrm{S_1S_0}}$ is saved to DIFF_DENSITY_S1_S0.cube file. When runtyp = cube, the log file (DIFF_DENSITY.log) and results directory (DIFF_DENSITY) are not generated. Error messages are written in DIFF_DENSITY_S1_S0.cube.

7. Calculate overlap (transition) density between $\mathrm{S_0}$ and $\mathrm{S_1}$ ($\rho_{\mathrm{S_0S_1}}$).
   A sample input file (RHO_0_3.inp) is as follows:

```
$comment
Input file for calculating overlap (transition) density
between S0 and S1
| 0 > : ground state (S0)
| 3 > : lowest excited singlet state (S1)
$end_comment

$control
property = rho
runtyp   = density
method   = td
$end_control

$elec_state
bra = < 0 |
ket = | 3 >
$end_elec_state

$grid
xmin = -5.0
xmax =  5.0
ymin = -5.0
ymax =  5.0
zmin = -5.0
zmax =  5.0
dx   = 0.20
dy   = 0.20
dz   = 0.20
$end_grid

DIR_INP_ELEC = ../INP_ELEC
```

To run d77, type:

```
d77 RHO_0_3
```

Figure 1 shows the ρ_{S_0}, ρ_{S_1}, $\Delta\rho_{\mathrm{S}_1\mathrm{S}_0}$, and $\rho_{\mathrm{S}_0\mathrm{S}_1}$ distributions visualized with Chemcraft.

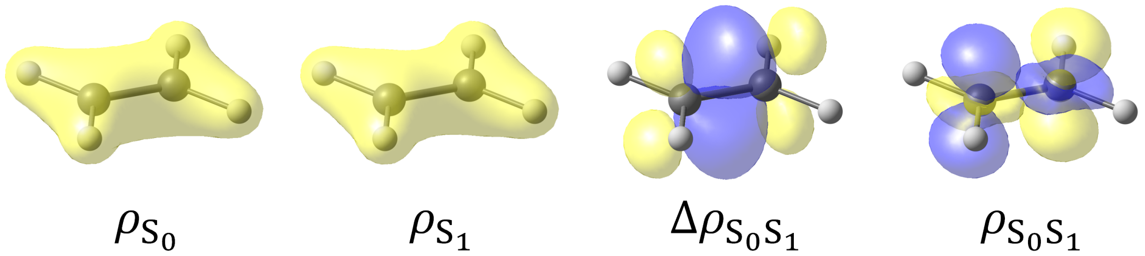

Figure 1: Electron densities of S_0 and S_1 (ρ_{S_0} and ρ_{S_1}, respectively), electron-density difference $\Delta\rho_{\mathrm{S}_1\mathrm{S}_0}$, and S_0-S_1 overlap density ($\rho_{\mathrm{S}_0\mathrm{S}_1}$): C_2H_4 as an example. Yellow shows positive; blue shows negative.

6.4 Transition dipole moment densities between S_0 and S_1 at optimized S_0 geometry

This example shows how transition dipole moment densities are calculated with d77. Sample d77 input and output files are in the examples/C2H4/TDMD directories. The first three steps are the same as in **6.1**.

1. Run geometry optimization and frequency analysis with Gaussian 16. 6D and 10F options are required to run d77.

2. Generate freq.fchk and TD.fchk files.

3. Run dpp.

4. Calculate transition dipole moment densities between S_0 and S_1.
   Create TDMD directory and move into it:

   ```
   mkdir TDMD
   cd TDMD
   ```

   A sample input file (TDMD_0_3.inp) is as follows:

   ```
   $comment
   Input file for calculating transition dipole moment density
   between S0 and S1
   | 0 > : ground state (S0)
   | 3 > : lowest excited singlet state (S1)
   $end_comment

   $control
   property      = dipole
   runtyp        = density
   method        = td
   save_rho_cube = yes
   $end_control

   $elec_state
   bra = < 0 |
   ket = | 3 >
   $end_elec_state

   $grid
   xmin = -5.0
   xmax =  5.0
   ymin = -5.0
   ymax =  5.0
   zmin = -5.0
   zmax =  5.0
   dx   = 0.20
   dy   = 0.20
   dz   = 0.20
   $end_grid

   DIR_INP_ELEC = ../INP_ELEC
   ```

To run d77, type:

```
d77 TDMD_0_3
```

d77 calculates the S_0-S_1 overlap density ($\rho_{S_0S_1}$), $-ex$, $-ey$, $-ez$, and the transition dipole moment densities for the x, y, and z directions (τ_x, τ_y, and τ_z). d77 also calculates the transition dipole moment densitiy (τ_{μ}) projected to the transition dipole moment vector ($\boldsymbol{\mu}$). The calculated $\rho_{S_0S_1}$, $-ex$, $-ey$, $-ez$, τ_x, τ_y, τ_z, and τ_{μ} are written to RHO.cube, ER_X.cube, ER_Y.cube, ER_Z.cube, TDMD_X.cube, TDMD_Y.cube, TDMD_Z.cube, and TDMD_MU.cube in the TDMD_0_3 directory. RHO.cube ($\rho_{S_0S_1}$) calculated in this example is the same as that calculated in **6.3**. You can skip generating RHO.cube by adding the following option in your control section:

```
save_rho_cube = no
```

The default setting is:

```
save_rho_cube = yes
```

Figure 2 shows the $\rho_{S_0S_1}$, τ_x, τ_y, and τ_z distributions visualized with Chemcraft. τ_x, τ_y, and τ_z are calculated from the relations: $\tau_x = \rho_{S_0S_1} \times (-ex)$; $\tau_y = \rho_{S_0S_1} \times (-ey)$; $\tau_z = \rho_{S_0S_1} \times (-ez)$. The spatial integral of τ_x, τ_y, and τ_z gives the x, y, and z components of the S_0-S_1 transition dipole moment $\boldsymbol{\mu}$: $\boldsymbol{\mu} = \left(\int \tau_x d\mathbf{r}, \int \tau_y d\mathbf{r}, \int \tau_z d\mathbf{r}\right)$. τ_{μ} is the transition dipole moment density projected to the direction of $\boldsymbol{\mu}$: $\tau_{\mu} = -\rho_{S_0S_1} \times e\mathbf{r} \cdot \boldsymbol{\mu}/|\boldsymbol{\mu}|$; $\int \tau_{\mu} d\mathbf{r} = |\boldsymbol{\mu}|$. The positive and negative regions of τ_x and τ_y are cancelled out and therefore, the x and y components of $\boldsymbol{\mu}$ vanish ($\int \tau_x d\mathbf{r} = 0$; $\int \tau_y d\mathbf{r} = 0$). Meanwhile, the negative region of τ_z is more widely distributed than the positive region, leading to non-zero $\int \tau_z d\mathbf{r}$: $\boldsymbol{\mu} = (0, 0, -3.31962)$ in debye.

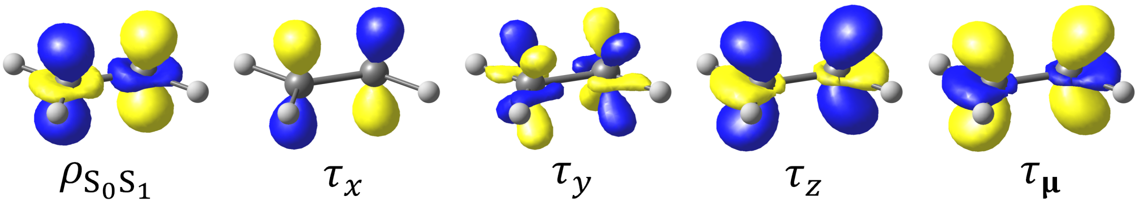

Figure 2: $\rho_{S_0S_1}$, τ_x, τ_y, τ_z, and τ_{μ} distributions. Yellow shows positive; blue shows negative.

6.5 Vibronic coupling densities between S_0 and S_1 at optimized S_0 geometry

This example shows how vibronic coupling densities are calculated with d77. Sample d77 input and output files are in the examples/C2H4/VCD directories. The first three steps are the same as in **6.1**.

1. Run geometry optimization and frequency analysis with Gaussian 16. 6D and 10F options are required to run d77.

2. Generate freq.fchk and TD.fchk files.

3. Run dpp.

4. Calculate vibronic coupling densities between S_0 and S_1.
   Create VCD directory and move into it:

   ```
   mkdir VCD
   cd VCD
   ```

   A sample input file (VCD_0_3.inp) is as follows:

   ```
   $comment
   Input file for calculating vibronic coupling densities
   for S0-S1 transition
   | 0 > : ground state (S0)
   | 3 > : lowest excited singlet state (S1)
   $end_comment

   $control
   property = vc
   runtyp   = density
   method   = td
   $end_control

   $elec_state
   bra = < 0 |
   ket = | 3 >
   $end_elec_state

   $vibmode
   7
   9
   $end_vibmode

   $grid
   xmin = -5.0
   xmax =  5.0
   ymin = -5.0
   ymax =  5.0
   zmin = -5.0
   zmax =  5.0
   dx = 0.10
   dy = 0.10
   dz = 0.10
   ```

```
$end_grid

DIR_INP_ELEC  = ../INP_ELEC
DIR_INP_VIB   = ../INP_VIB
DIR_INP_ELFLD = ../ELFLD
```

To run d77, type:

```
d77 VCD_0_3
```

d77 calculates the S$_0$-S$_1$ overlap density ($\rho_{\mathrm{S_0S_1}}$), the derivatives of the nuclear electronic potentials for the seventh and ninth vibrational modes (ν_7 and ν_9), and the vibronic coupling densities for the seventh and ninth vibrational modes (η_7 and η_9). The calculated $\rho_{\mathrm{S_0S_1}}$, ν_7, ν_9, η_7, and η_9 are written to RHO.cube, DVNE0007.cube, DVNE0009.cube, VCD0007.cube, and VCD0009.cube files in the VCD_0_3 directory. RHO.cube ($\rho_{\mathrm{S_0S_1}}$) calculated in this example is the same as that calculated in **6.3**. You can skip generating RHO.cube by adding the following option in your control section:

```
save_rho_cube = no
```

Figure 3 shows the $\rho_{\mathrm{S_0S_1}}$, ν_7, ν_9, η_7, and η_9 distributions visualized with Chemcraft. ν_7 and ν_9 dsitributions reflect the seventh and ninth vibrational modes, respectively. η_7 and η_9 are calculated from the relations: $\eta_7 = \rho_{\mathrm{S_0S_1}} \times \nu_7$; $\eta_9 = \rho_{\mathrm{S_0S_1}} \times \nu_9$. The spatial integral of η_7 and η_9 gives VCCs for the seventh and ninth vibrational modes, respectively.

The η_7 distribution suggests that the vibronic coupling for the seventh vibrational mode occurs around the carbon atoms. The η_9 distribution suggests that the vibronic coupling for the ninth vibrational mode occurs along the C−H bonds.

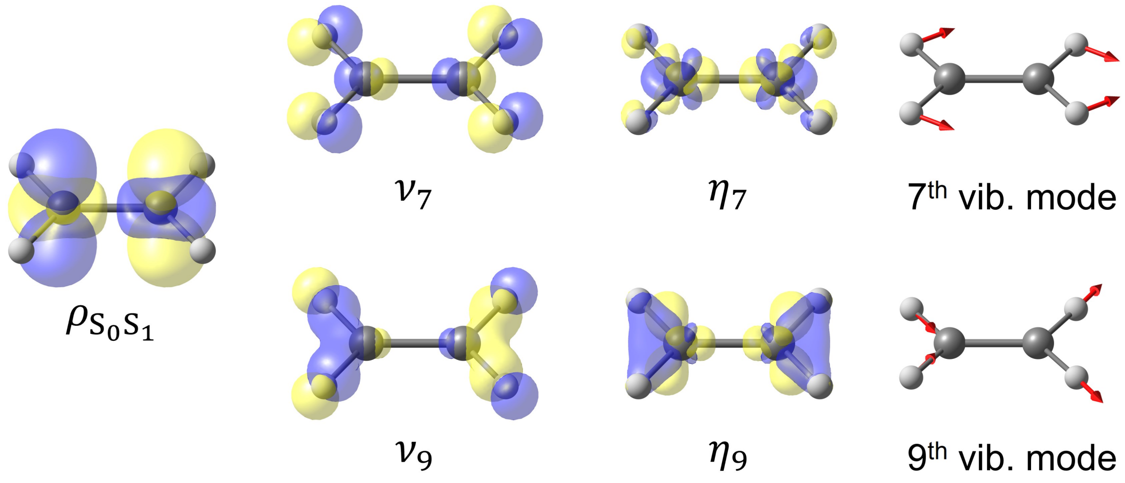

Figure 3: $\rho_{\mathrm{S_0S_1}}$, ν_7, ν_9, η_7, and η_9 distributions. Yellow shows positive; blue shows negative.

References

[1] d77: visual understanding of molecular properties. Version 20240427. https://github.com/KatsuyukiShizu/d77.

[2] L. E. McMurchie and E. R. Davidson, Journal of Computational Physics **26**, 218 (1978).

[3] S. Koseki, M. W. Schmidt, and M. S. Gordon, The Journal of Physical Chemistry **96**, 10768 (1992).

[4] S. Koseki, M. S. Gordon, M. W. Schmidt, and N. Matsunaga, The Journal of Physical Chemistry **99**, 12764 (1995).

[5] S. Koseki, M. W. Schmidt, and M. S. Gordon, The Journal of Physical Chemistry A **102**, 10430 (1998).

[6] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. V. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery, Jr., J. E. Peralta, F. Ogliaro, M. J. Bearpark, J. J. Heyd, E. N. Brothers, K. N. Kudin, V. N. Staroverov, T. A. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. P. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman, and D. J. Fox, Gaussian 16 Revision C.01, 2016, gaussian Inc. Wallingford CT.

[7] Chemcraft - graphical software for visualization of quantum chemistry computations. Version 1.8, build 682. https://www.chemcraftprog.com.

[8] R. Dennington, T. A. Keith, and J. M. Millam, GaussView Version 6, 2019, semichem Inc. Shawnee Mission KS.